УДК 004.424

ПАТТЕРН MVVM В РАЗРАБОТКЕ ПРИКЛАДНЫХ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ ДЛЯ ПЛАТФОРМЫ ANDROID

Кутовенко А.А.

БНТУ, Минск, Беларусь, kutovenko@gmail.com

Опора на рекомендуемую архитектуру MVVM при обучении разработке мобильных приложений на платформе Android, создании соответствующих учебно-методических материалов обладает рядом преимуществ. Студенты получают опыт работы со стандартными современными решениями, получают навык универсального проектирования структуры мобильного приложения. Наличие официальных примеров облегчает разработку учебно-методических материалов.

Одна из наиболее сложных задач в обучении объектно-ориентированному программированию является обучение проектированию архитектуры приложений. Практические задачи могут быть решены различными способами. Рабочее приложение можно получить и применив не самую эффективную архитектуру. Подобное разнообразие затрудняет получение студентами представлений о лучших практиках решения сложных задач.

В ситуации обучения студентов программированию мобильных приложений ситуация усложняется, поскольку к таким приложениям дополнительно предъявляется ряд требований, отсутствующих на традиционных платформах. В частности, это жесткие требования к экономному энергопотреблению. Помимо этого, для работы с мобильными приложениями характерен стиль быстрого просмотра приложений, постоянных включений и выключений дисплея. Выполнение любого приложения на смартфоне может быть внезапно прервано телефонным звонком. Вследствие этого на платформе Android находящееся в фоне приложение в любой момент может быть закрыто операционной системой при нехватке ресурсов для работы других программ. Все подобные ситуации приходится учитывать. Для этого на платформе Android служит ряд методов в стандартных классах Activity и Fragment, относящихся к различным стадиям жизненного цикла приложения и позволяющих корректно обрабатывать названные ситуации. Однако, подобные задачи нетривиальны для начинающих разработчиков, достаточно трудоемки и могут выступать источником ошибок.

В то же время, в промышленном проектировании давно используются понятие "паттерн", которое соответствует найденному и постоянно применяемому способу решения какой-либо типовой проблемы в разработке приложений. Паттерны, как правило, не охватывают архитектуру приложения целиком, а относятся к способам создания объектов (порождающие паттерны), эффективной внутренней организации классов (структурные паттерны), проектированию их методов (поведенческие паттерны). Однако, есть и исключения. В частности, это паттерн Model-View-View Model, который позиционируется корпорацией Google в качестве универсальной типовой модели для разработки современных мобильных приложений. Данный паттерн совершенствует классическую архитектуру MVC и позволяет еще на стадии проектирования решить ряд проблем, специфичных для мобильных приложений, в частности, проблему управления событиями жизненного цикла компонентов приложения [1].

Паттерн MVVM состоит из нескольких структурных компонентов. Для его поддержки корпорация Google выпустила ряд готовых библиотек, предназначенных для решения часто

возникающих задач. Кратко рассмотрим содержание названных компонентов.

Компонент Model представляет основную бизнес-логику приложения. Именно здесь организуется связь с источниками данных: СУБД, сетевыми ресурсами. Классы данного компоненты строятся с использованием классического паттерна "Наблюдатель" (Observer), позволяющего автоматически обновлять пользовательский интерфейс при изменениях в источниках данных. В целом назначение данного компонента совпадает с назначением компонента Model в привычной архитектуре MVC.

Компонент View Model является специфическим для мобильных приложений. Он позволяет построить дополнительный уровень абстракции между моделями данных и элементами пользовательского интерфейса приложения. Он готовит и хранит данные, выводимые в пользовательском интерфейсе. Именно наличие независимого компонента View Model позволяет обрабатывать упомянутые ранее ситуации перестройки объектов, внезапных прерываний выполнения приложений, не теряя пользовательские данные с помощью более простого и компактного кода, чем при традиционной обработке событий жизненного цикла мобильного приложения.

Компонент View выступает наблюдателем по отношению к компоненту View Model, получает от него данные и обновляет соответствующие элементы пользовательского интерфейса.

Наличие официально предлагаемой архитектуры упрощает обучение проектированию мобильных приложений. Создавая теоретические материалы и практические задания, следует сначала делать акцент на демонстрации универсальности предлагаемого паттерна, а по мере его освоения переносить акцент на особенности реализации в различных типовых задачах. В решении такой методической задачи помогает наличие официального репозитория [2], содержащего примеры проектов, использующих новые типовые MVVM и их взаимосвязь при решении ряда типовых задач: компоненты построению интерфейса приложения, взаимодействия с базами данных, сетевыми сервисами. Наличие таких примеров облегчает разработку практических заданий, а также последовательное освоение MVVM.

Список литературы:

1. Android Jetpack [Electronic Resource] / Google Developers . - Mode of Access: https://developer.android.com/jetpack/ . - Date of Access: 10.11.2018.

2. Samples for Android Architecture Components [Electronic Resource] / Google Developers. - Mode of Access : https://github.com/googlesamples/android-architecture-components . - Date of Access: 10.11.2018.